# 算术表达式计算器的设计流程和算法思路

（请关闭杀毒软件后运行程序）

## 基本算法思路：

1. 输入一个算术表达式，算术表达式输入类型为string型；
2. 将输入的算术表达式进行分块，例如：输入的算术表达式为1\*2+3，则分块之后将变成1，\*，2，+，3。使用链表进行存储，链表的数据域为数据本身；在分块过程中，进行非法字符的检查和非法小数，乘方的检查。假如检查到非法字符，则这储存这个非法字符的节点存储的数据域为一个FALSE指标；
3. 对分块之后的链表进行遍历，假如遍历到有一个节点的数据域为数据+FALSE，则直接输出“输入有误，请重新输入算术表达式”
4. 假若分块之后的链表中的数据没有不合法，进行遍历判定输入算术表达式的格式合法性，判定指标包括：开头是否为算术表达式，左右括号数是否相同，是否存在两个算术符号相邻的情况。
5. 假如算术表达式输入格式正确，利用栈运算进行算术表达式中缀表达式求值算法，即边进栈，边运算。同时将另外两个模拟栈用作实时输出栈内元素和实时具体栈操作；

## 重要函数设计思路

//将输入的算术表达式分解，每一部分存储成链表的节点的形式

函数Linkstring\* transform(string str){}；

1. 从字符串的头字符开始遍历；
2. 当该位置字符为+，-，\*，/，（，）时，记录成一个节点；
3. 当该位置字符为数字时，判断下一位是否为数字或者小数点，或者“^”，假如是，一直加到一个字符串的末尾，直到下一位不是数字或者小数点，或者“^”，最后的字符串记录成一个节点的数据；
4. 当该位置为非法字符时，节点数据记为FALSE；

检验输入是否出错的函数

bool correction(Linkstring\* str)

1. 遍历传进来的链表，判断是否存在节点的数据为FALSE，若存在，则直接返回0，结束该函数；否则继续检验；
2. 判断开头首字符是否为符号，若是符号，则直接返回0；结束该函数，否则继续检验；
3. 当为右括号时，右括号计数器+1，其前一位不能为除数字和右括号之外的；若不符合，judge=0；结束遍历；
4. 当为左括号时，左括号计数器+1；其后一位数不能为除数字和右括号之外的；若不符合，judge=0；结束遍历；
5. 相邻两个节点的数据类型不能出现+，-，\*相邻的情况；若出现，judge=0；结束遍历；
6. 当最后一个字符为+，-，\*，/，judge=0；
7. 如果右括号数不等于左括号数，judge = 0 ；
8. 返回judge；

Void run()函数模拟算术表达式中缀模拟运算的过程；

long double transform\_to\_num(string s)就是将字符串类型存储的数字转换成double型；